Project MAC, an M.I.T.
and Research Projects
Office of Naval Research
the primary support for
, departments and labora-
ose research programs are,
it and private agencies.
: mention of agency and
on

in part, is permitted for
me t. Distribution of this

es of this Progress Report,
listed in Appendix B, from
ati , Alexandria, Virginia,
qu ts are submitted on
able from the office in your
he focal point for requesting

ay obtain copies of this
era Scientific and Technical
5 Port Royal Road, Springfield,

he cover
ctions of a
ph plotted
phics facility.

PROGRESS REPORT VI

JULY 1968 to JULY 1969

PROJECT MAC

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

545 Technology Square Cambridge, Massachusetts 02139

JUN 2 9 1970

Center for Computer
Sciences and Technology

NBS 487/10

PROJECT MAC ADMINISTRATION

Prof. J. C. R. Licklider	Director
Prof. M. M. Jones	Assistant Director
David E. Burmaster	Assistant Director for Student Activities and Business Manager
Dorothea C. Scanlon	Assistant to the Director
H. E. Brammer	Assistant to the Director (to June 1969)
M. L. Cabral	Business Manager (to March 1969)

L. S. Cavallaro
D. A. Duggento
R. C. Goldstein
C. D. Graceffa
J. M. Greene
J. A. Gunn
D. Kontrimus
R. G. Mansfield
L. L. Maddaus
E. G. Moore
L. G. Pantalone
E. M. Roderick
R. T. Stetson
C. M. White
E. M. Wolman
M. L. Young

INTRODUCTION

Project MAC is an M.I.T. interdepartmental research laboratory for computer science and engineering. The research program of Project MAC is focused upon the new field of "interactive" computing, i.e., computing in which men and computers work together in close partnership in solving problems or making decisions. Some of the research is aimed at making it possible for men and computers to work together more effectively. Some of it takes advantage of interactive computing to facilitate the solution of basic problems in computer science or to develop new applications. But almost all of it deals with men and computers -- hence the acronym "MAC". (There are other expansions of "MAC"; those we use most often are "multiple-access computers" and "machine-aided cognition".)

During the last year, Project MAC has accomplished several research objectives, and it has gone through part of a major transition. Inasmuch as a brief account of the transition will set the stage for a discussion of the research results, let us begin with that account.

The Course of Project MAC

In order to bring about a partnership between men and computers, one must make it possible -- and economically practicable -- for men and computers to work together directly and effectively. The first major undertaking of Project MAC, beginning in 1963, and building on early work of the M.I.T. Computation Center, was to create a computer system with which many people could work simultaneously and conveniently. The result was the first general-purpose multi-access computer system, CTSS, the Compatible Time-Sharing System. (Actually, two CTSS's were built; one was operated by Project MAC, the other by the Computation Center.)

By 1965, CTSS was the focus of a significant part of the intellectual effort of about 200 research people at M.I.T. and quite a few at other universities. It was obvious that interactive computing opened new horizons. The present computer "time-sharing" industry, with its 150 to 200 service companies, is one of them. CTSS made it evident that a multi-access computer could provide the communication as well as the information storage and processing facilities required for the emergence of a new kind of intellectual community. Partly by having them and partly by lacking them, CTSS made it possible to see what characteristics and features a "community" multi-access system should possess.

The effect of CTSS upon Project MAC was threefold: Its quick success, together with the flooding-in upon the designers of ideas

about how to build a far better system, led to the determination to create a truly comprehensive community computer system, and, with the General Electric Company and the Bell Telephone Laboratories, Project MAC initiated a major effort toward that goal. Not all the computer scientists and engineers who contributed to the development of CTSS wanted to participate in the development of the more advanced system: many turned to research in other aspects of computer science and engineering. At the same time, CTSS tended to bring its community of substantive users -- people interested in using interactive computing to facilitate research in their various fields -- into Project MAC. Thus Project MAC came to have three main parts: (1) the Computer System Research Group, developing the new multi-access computer system, (2) several smaller research programs in computer science and engineering, and (3) a large number of users of CTSS.

During 1967 and 1968, as the use of CTSS turned from a research experience into an operational routine, and as the development of the new multiple-access system required an increasing fraction of the available funds, the third part dwindled. This past year, the CTSS operated by Project MAC was transferred to the Information Processing Center (formerly Computation Center), and Project MAC became a two-part laboratory. That was the first part of the transition.

The second part of the transition began on the first of January 1969. The new multi-access system, called the Multiplexed Information and Computing Service (Multics), had proven to be much more complex and difficult than anticipated, and its development had proceeded at just half the scheduled speed -- until the beginning of the new year. Then everything began to go well, the schedule quit slipping and now (July 1969) the heart of Multics beats, the Multics Operating System operates.

The Multics that now "runs" is a bare-bones system, but an advanced one. The Computer System Research Group is confident that it will be opened for general use on 1 October 1969. That will begin the third part of the transition. The main system research effort will turn from the operating system to the comprehensive library procedures and data required to convert such a "system" into a "service", to make it effectively and conveniently useful to others than computer buffs. This service-building effort will lead, we believe, to a realization of our now long-dreamed dream of an "on-line" intellectual community. But it will not bring the whole community into Project MAC -- only the service builders. Operational responsibility for Multics will be transferred to the Information Processing Center.

Characteristics and Features of Multics

Inasmuch as the demonstration of the Multics Operating System was one of the main accomplishments of the past year, it is an appropriate time to explain the basis for the claim that Multics provides a new and unique facility for a true on-line community of users, a foundation for unprecedented teamwork in many kinds of undertaking that are based upon information. There are two main areas in which Multics represents a major advance. In the first of these areas, a few other systems, developed since the design of Multics was published, are comparable to Multics. In the second, to the best of our knowledge, Multics is unique.

First, Multics introduces a great simplification into the individual user's concept of the computer system. Heretofore, he had not only to remember the names of his programs and sets of data but also to keep in mind their sizes and, if large, squeeze them (all together or in subsets) into the computer's memory, which was pictured as a single series of little bins, each capable of holding one "word" of procedure or data. Using Multics, he thinks only in terms of the names; the operating system brings into memory those procedures and data required at each moment and does not waste expensive memory space (which can be put to good purpose by other users) on a user's inactive information. Programming within the context of Multics, the user has available to him the equivalent of millions of pieces of writing paper, each just as long as he needs it to be -- but he need not remember how long, or where he put the pieces; he just names them and remembers the names. (If he forgets the names, Multics will of course provide a listing.) Nor does the user have to think about the configuration of the "hardware" computer. It may change basically, e.g., in number of processing units or in number of memory blocks, through failure of subsets or through augmentation, but the only way that affects the user is to slow down or speed up the service. His working image of the configuration of the computer system remains the same: a "space" occupied by the names of procedures and data.

Second, Multics greatly facilitates cooperation among its users. It makes it easy for them to work in pairs or groups, using shared as well as individual programs and data sets and communicating with one another through their consoles as well as face-to-face or by telephone. To use another person's procedure or data, one has only to get his permission. One does not have to borrow a copy either overtly or within the system; two or more people can use the same "copy" of a procedure or of a set of data at the same time. Multics keeps one of them (and his programs) from changing it while the other (another) is in the process of "reading" it. Without permission of the owner, access to files is barred. The owner

can extend permission to any individual or established group to use one of his files in any one of several ways: "read and change", "read only", "execute only", and "use only through a privileged program". The Multics file system provides the basis for free, voluntary cooperation, and also the basis for entrepreneurial co-operation for a fee: Multics will keep records of the use of certain programs and data and send bills on behalf of the owner.

Professor F. J. Corbató, leader of the Computer System Research Group, has a list of 30 characteristics and features, of which the foregoing are a few, that define Multics. It will not be long, now, until we see whether they will make the expected difference, whether they will greatly facilitate the individual's use of his own information and, at the same time, turn the computer into a communication network.

Knowledge and Heuristics in the Computer

Computers are so fast and accurate in their execution of procedures that they proved themselves very useful long before any computer was programmed to do anything that seemed at all intelligent. Indeed, to combine the computer's speed and accuracy in applying defined procedures to specified data with man's ability to formulate and evaluate is the essential aim of man-computer interaction. But communication between men and ordinary computer systems is so poor as to frustrate any attempt at teamwork. There can be little man-computer partnership if the computer knows practically nothing and has to be told at every stage precisely what to do and in detail how to do it.

A large and important part of the work of Project MAC is devoted, therefore, to learning how to "educate" the computer to be a better partner. It does no good simply to fill the computer's memories and stores with facts; the uneducated computer can remember but not use them. It does little good to fill the memories and stores with procedures; the uneducated computer has to be told exactly when and to what data to apply each one. The problem is, essentially, to understand the process of being intelligent and to program at least some of that process into the computer. It is clear that the process involves knowledge, which can be defined roughly as facts structured into a model that can be "run" or be interpreted by the processor, and heuristics, which can be defined roughly as guidelines to solution or discovery.

In Project MAC, during the past year, several accomplishments were made in the education of computers. In most instances, the computer system was the Incompatible Time-Sharing System (ITS), a system somewhat smaller than CTSS and Multics and specialized in quite different ways. You have to be a computer buff to use

ITS -- but, if you are, you like it so much that you do nothing to make it easy for non-buffs to understand. As a result of the educational accomplishments, ITS can -- among other things -- now:

1) Solve calculus "word problems" of the type, "A man 6.0 feet tall walks at the rate 5.0 feet per second toward a street light which is 16.0 feet above the ground. At what rate is the tip of his shadow moving?" (But see Sec. 1 of this Report on Artificial Intelligence and Intelligent Automata for a discussion of difficulties with similar problems.)

2) Look (through its television-camera "Eye") at a haphazard pile of toy blocks on the floor, analyze the scene into individual blocks (Prof. A. Guzman; B. K. P. Horn), pick them up one-by-one (with its mechanical arm and hand), and stack them up to make a tower.

3) Play Class C chess instead of Class D chess, as last year. (R. Greenblatt)

4) Solve symbolic (non-numeric) integration problems, now even including problems involving logarithms and exponentials, such as the Gaussian ("error") function. (Prof. J. Moses)

The systems of programs that do the foregoing things, as well as others now operating, have some knowledge and some heuristic capability, but not much. They are successful only in quite special and restricted contexts. A large group, led by Professors M. M. Minsky and S. A. Papert, is working intensively to advance the understanding of such problems.

Man-Computer Interaction

Next to the stupidity of ordinary computer systems, the main barrier to effective and convenient communication between men and computers has been the computer "console". In practical fact, the console used in 98 per cent of all on-line man-computer communication these last few years has been merely an electric typewriter with additional electromechanical parts to send codes to and receive codes from a computer. Such typewriters are narrow bottlenecks. Obviously they had to give way to, or at least be supplemented by, graphic displays and non-keyboard computer-input devices such as pencils and microphones.

The past year saw a marked advance in the commercial supply of ultra-typewriter equipment for man-computer interaction. The Advanced Remote Display Station (ARDS), developed by the Electronic Systems Laboratory in conjunction with Project MAC, led an important part of the advance. Its storage cathode-ray screen provides a non-flickering image of fairly good resolution and the capacity to

and other problems. We
o tacles. The image-
i...ages (at the expense of
ses, was unimportant) so, in
on , Inc., we designed and
n image dissector is es-
an image projected on a
are focused through a de-
m selected image point
mc...sured.

atio depends essentially on
bt n more precision at the
cic...d to put this signal-to-
ol and to make it independent
as ed. This is built into
ve the details ("The Image
the constant signal-to-noise
dim points, the exposure
sy: em has also a dark cut-
ur nent very early if an
below a specified value. This
e sulting system can meas-
ng with a brightness-
64 throughout this range.
number or as a logarithm
nc brightness input that
a...ost independent of over-
re various protective de-
ub by excessively bright
m: y advantages and pitfalls
ing using image dissectors
with us about the results of
io:

to build various forms of
rhaps along the lines of
f the vertebrate retina and
nearly so well understood
think the time is quite ripe
ou present operations, the
vels of scene-analysis
-level pre-processing, so at
ctor to be gained from fast
le to decide how it should

References

1. Lawrence G. Roberts, Machine Perception of Three-Dimensional Solids, Department of Electrical Engineering, M.I.T., Ph.D. Thesis, May 1963.

2. Thomas G. Evans, "A Program for the Solution of Geometric-Analogy Intelligence Test Questions", Semantic Information Processing, M.I.T. Press (Cambridge) 1968, pp. 271-353.

3. A. Newell, "Learning, Generality and Problem Solving", Information Processing 1962, IFIP Congress 1962, North Holland Publishing Co. (Amsterdam) 1963, pp. 407-412.

4. Edward A. Feigenbaum, "The Simulation of Verbal Learning Behavior", Computers and Thought, McGraw-Hill Book Co. (New York), 1963, pp. 297-309.

5. M. A. K. Halliday, "Some Notes on 'Deep' Grammar", J. Linguistics, 3, 1967.

6. Terry A. Winograd, "Linguistics and the Computer Analysis of Tonal Harmony", J. Music Theory, 12, 1968.

7. R. Anderson, "Syntax-Directed Recognition of Hand-Printed Two-Dimensional Mathematics", Ph.D. Thesis, Division of Engineering and Applied Physics, Applied Math, Harvard University, Cambridge, Mass., Jan. 1968.

8. H. A. Ernst, MH-1, A Computer-Operated Mechanical Hand, Department of Electrical Engineering, M.I.T., Ph.D. Thesis, December 1961.

Some Relevant Internal Memos
of the Artificial Intelligence Group

101 SIDES 21, R. D. Greenblatt, J. T. Holloway; described by Donald Sordillo, August 1966, MAC-M-320.

123 Computer Tracking of Eye Motions, M. L. Minsky, S. A. Papert, March 1967 (Vision).

131 POLYSEG, A. K. Griffith, April 1967 (Vision).

145 A Fast-Parsing Scheme for Hand-Printed Mathematical Expressions, W. A. Martin, Oct. 1967, MAC-M-360.

160 Focusing, B. K. P. Horn, May 1968.

161A ITS 1.5 Reference Manual, D. E. Eastlake III, MAC-M-377. Revised July 1969 (ITS 1.4 Ref. Manual June 1968).

163 Holes, P. H. Winston, Aug. 1968.

ARTIFICIAL INTELLIGENCE

164 Producing Memos using TJ6, TECO and the Type 37 Teletype, L. J. Krakauer, Sept. 1968.

165 Description and Control of Manipulation by Computer-Controlled Arm, J.-Y. Gresser, Sept. 1968.

166 Recognition of Topological Invariants by Modular Arrays, W. T. Beyer, Sept. 1968.

168 PLANNER, C. E. Hewitt, MAC-M-386. Oct. 1968; revised June 1969.

169 PEEK and LOCK, D. E. Eastlake III, MAC-M-387, Nov. 1968. Replaced by revised Memo 161A, July 1969.

171 Decomposition of a Visual Scene into Three-Dimensional Bodies, Adolfo Guzman, Jan. 1961, MAC-M-391.

172 Robot Utility Functions, Stewart Nelson, Michael Levitt, Feb. 1969 replaced by Revised Memo 161A, July 1969.

173 A Heuristic Program that Constructs Decision Trees, March 1969, P. H. Winston.

174 The Greenblatt Chess Program, R. D. Greenblatt, D. E. Eastlake III, Stephen Crocker, April 1969.

175 On Optimum Recognition Error and Reject Tradeoff, C. K. Chow, April 1969.

176 Discovering Good Regions for Teitelman's Character Recognition Scheme, P. H. Winston, May 1969.